THE UNIVERSITY of EDINBURGH

Edinburgh Research Explorer

# Reasoning about representations in autonomous systems: what Pólya and Lakatos have to say

**Citation for published version:**
Bundy, A 2012, Reasoning about representations in autonomous systems: what Pólya and Lakatos have to say. in D McFarland, K Stenning & M McGonigle-Chalmers (eds), *The Complex Mind: An Interdisciplinary Approach.* Palgrave Macmillan, pp. 167-183. https://doi.org/10.1057/9780230354456_9

**Digital Object Identifier (DOI):**
10.1057/9780230354456_9

**Link:**
Link to publication record in Edinburgh Research Explorer

**Document Version:**
Early version, also known as pre-print

**Published In:**
The Complex Mind: An Interdisciplinary Approach

**General rights**
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

**Take down policy**
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 04. May. 2023

# Reasoning about Representations in Autonomous Systems: What Pólya and Lakatos have to say.*

Alan Bundy

August 14, 2012

## Introduction

Autonomous reasoning systems combine an often logic-based representation of some aspect of the world with rules for manipulating that representation. These representations are usually inherited from the literature or are built manually for a particular reasoning task. They are then regarded as fixed. We have argued that representations should instead be regarded as fluid, i.e., their choice, construction and evolution should be under the control of the autonomous agent rather than predetermined and fixed [Bundy & McNeill, 2006].

1. Appropriate representation is the key to successful problem solving. It follows that a successful problem solver must be able to choose, construct or evolve whatever representation is best suited to solving the current problem.

2. Autonomous agents use representations called *ontologies*. For different agents to communicate they must align their ontologies. In some applications it is not practical to manually pre-align the ontologies of all agent pairs – it must be done dynamically and automatically.

3. Persistent agents must be able to cope with a changing world and changing goals. This requires evolving their ontologies as their problem solving task evolves. The W3C call this *ontology evolution*[1].

George Pólya has written the classic guide to the art of problem solving [Pólya, 1945]. Imre Lakatos has written a fascinating rational reconstruction of the evolution of mathematical methodology [Lakatos, 1976]. Although it was not their intention to do so, both these authors have implicitly provided profound evidence for our thesis that representations should be fluid. In this paper we analyse their work and extract this evidence.

## Pólya's How to Solve It

George Pólya (1887-1985) was a world famous mathematician who did seminal work in series, number theory, mathematical analysis, geometry, algebra, combinatorics, and probability. He is, however, best known for his books explaining the art of problem solving to students of mathematics. The most popular these books is his "How to solve it" [Pólya, 1945]. The backbone of this book is a list of questions[2] that problem solvers are urged to ask themselves to assist their creativity. Many commentators have encouraged automated reasoning researchers to apply Pólya's questions in the construction of autonomous agents.

*The research reported in this paper was supported by EPSRC grant EP/E005713/1. I would like to thank the members of the Mathematical Reasoning Group who gave me feedback, especially Michael Chan and Alison Pease. Especial thanks for Predrag Janičić and Alison Pease for drawing the images.

[1] `http://www.w3.org/TR/webont-req/#goal-evolution`

[2] A few are not questions, but injunctions.

Surprisingly, despite widespread knowledge of and enthusiasm for Pólya's ideas, they have had very little influence on practical automated reasoning systems. In this paper, we will argue that this is because the nature of the advice has been widely misinterpreted. The main focus of automated reasoning research has been on search control, i.e., on mechanisms for choosing which reasoning steps to take amongst those that are applicable. But Pólya has relatively little to say about this. Rather, most of his advice is about *problem representation*[3].

To defend this claim, below I analyse each of Pólya's questions in turn. We will see that while a few of them *do* relate to proof search, most are about the choice, construction or evolution of problem representations. Pólya organised his questions into four phases: understanding the problem, devising a plan, carrying out the plan and looking back. I've kept these phases, as well as his grouping of several questions into a paragraph and his use of italics.

## Understanding the Problem

Understanding the problem is the initial phase in Pólya's suggested approach to mathematical problem solving. As its title suggests, this phase has little to do with proof search and much more to do with representation formation. Few autonomous reasoning systems automatically form a problem representation. Usually, the representation of a problem is manually encoded as an axiomatisation within some logic. Standard sets of such axiomatisations are shared by the research community, which uses them to evaluate and compare automated proof systems. These sets are under constant development, as new axiomatisations are added and old ones amended, but this development is typically done off-line by human researchers, not by automated autonomous agents[4]. The main exception to this rule is where an automated reasoning system is part of a question-answering system, and natural language processing is used to translate written or spoken utterances, in say English, to and from a logical representation. Even here, the underlying formal representation is usually manually encoded, and only additional facts added to it, translated from the input utterances.

Below, each of Pólya's questions is reproduced in bold, followed by some analysis.

### ***What is the unknown?***

By *unknown*, Pólya presumably means the value to be discovered as a result of problem solving. The problem input to a logic-based prover usually comes with a well-labelled goal or conjecture. For instance, it might be the instantiation to be found for $x$ in a problem of the form $\exists x. \ldots$. So it is not clear what addressing this question would add to the search for a proof. However, in representing an informally stated problem it is vital to identify what is given and what is sought, so we know what can be assumed and what has to be proved.

### ***What are the data?***

Similar remarks apply. By *data*, Pólya presumably means the various objects defined by terms in the conjecture. For instance, in a geometry problem, these might form the initial diagram. These are already known in a formally stated problem, but need to be identified when representing the problem. In fact, deciding how to formally represent informally described objects is called *idealization*, and is the critical issue in representation formation [Bundy, 2006]. For instance, in a relative velocity mechanics problem, a ship might be idealized as a point on a horizontal plane, but in a specific gravity problem it might be idealized as a container with volume but indeterminate shape.

[3]Which can also require search, but not *proof* search.

[4]Sutcliffe and Puzis have automated the *selection* of relevant axioms from a larger set, but their system does not create *new* axioms [Sutcliffe & Puzis, 2007].

***What is the condition?***

Again, similar remarks apply. By *condition*, Pólya presumably means $P$ in a conjecture of the form $P \Rightarrow Q$. Again, such conditions are identified by simple inspection in a formally stated problem. When representing the problem, however, just deciding that a conjecture should take the form $P \Rightarrow Q$ is a significant decision.

**Is it possible to satisfy the condition? Is the condition sufficient to determine the unknown? Or is it insufficient? Or redundant? Or contradictory?**

Checking that the condition is satisfiable or contradictory does sometimes form part of the proof search; if the condition is contradictory the conjecture is trivially true. Contradictory conditions can arise, for instance, when a conjecture is over-generalised or when several overlapping case splits are made.

However, checking that the condition is (in)sufficient to determine the unknown is not relevant when a formalised problem is to be *solved*, unless as a sanity check that you are not being made a fool of. It makes much more sense as a sanity check after *formalising* a problem to ensure this has been done in a sensible way. All the other checks can also be more usefully interpreted in this way.

**Draw a Figure. Introduce suitable notation.**

"Introduce suitable notation" is a complete give-away here. That's *exactly* what you are doing in forming a representation. Unless your proof search includes notational changes or augmentations, it does not make sense as a proof search hint. Humans often use a figure as a way of clarifying a problem during representation formation — or the diagram may *be* the representation. Figure drawing is not usually a part of automated *proof*, although there has been some work on the automation of reasoning with diagrams [Jamnik *et al*, 1997, Winterstein *et al*, 2002].

**Separate the various parts of the condition. Can you write them down?**

"Can you write them down?" seems to be an instruction to formalise the condition; prior separation being intended to break this difficult task into more manageable portions. Neither sentence makes much sense from a proof-search viewpoint, but both make sense from representation-formation viewpoint.

## Devising a Plan

On the face of it, devising a plan sounds very relevant to proof search. My group has developed a search control technique we call *proof planning*, in which an outline of a proof is used to guide proof search [Bundy, 1991]. Indeed, proof-search advice is *part* of this grouping of Pólya's advice. However, on closer examination, advice in this group is neither solely nor mainly about proof search. Pólya's plans include representation formulation as well as proof search. Indeed, his idea is to minimise proof search by careful representation. So, again, it is representation that is the central focus.

**Have you seen it before? Or have you seen the same problem in a slightly different form?**

This question suggests using the proof of an analogous theorem as a guide to constructing the proof of the current theorem. There has been periodic work using this technique. [Owen, 1990] provides a good survey up to 1990. However, such work has seldom been sustained and has not been very influential. So, this question is ostensibly a proof-search suggestion.

However, if you listen carefully to uses of analogy in everyday conversation, you will see that it is rarely using an old argument to guide the search for a new one. Rather it is suggesting a

way of *representing* the problem in a way analogous to an existing representation. The solution is thereby immediately suggested, i.e., no search is required. Consider, for instance, the following quote from King James I of England and VI of Scotland on taking the English throne in 1604 and arguing, in parliament, for the union of the two kingdoms.

> "I am the Husband and the whole Isle is my lawfull Wife; I am the Head; and it is my Body; I am the Shepherd and it is my flocke; I hope therefore no man will be so unreasonable as to think that I that am a Christian King under the Gospel should be a polygamist and husband to two wives; that I being the Head should have a divided and monstrous Body."

Note that the argument establishes an analogy (or rather three) and then imports the undesirability of properties of the analogy (polygamy, monsterousness) back to properties of the original situation (two kingdoms under one King) to assert their undesirability there. There is no demonstration that arguments (proofs) in the analogy translate into arguments in the original. Rather one is invited to import a representation for the new, two-kingdoms situation by translating an old representation for polygamy, etc., complete with a pre-formed conclusion about the undesirability of this situation.

***Do you know a related problem?*** **Do you know a theorem that could be useful?**

The first question above seems to say the same thing as the previous questions, and similar remarks apply. The instruction to find a useful theorem, however, is a rare example of a genuinely proof-search hint.

***Look at the unknown!*** **And try to think of a familiar problem having the same or a similar unknown.**

Again similar advice to before, and similar remarks about it. This time you are advised to use the unknown as a key and cue for the analogous problem.

***Here is a problem related to yours and solved before. Could you use it?*** **Could you use its result? Could you use its method? Should you introduce some auxiliary element in order to make its use possible?**

More of the same, but now you have the related problem and are invited to recycle its proof. This really *does* seem to be proof-search advice. You are advised either to use this theorem as a lemma in your current problem or to try to map its proof onto your proof using the kind of analogy mapping discussed in [Owen, 1990]. Pólya even draws attention to the need to patch the partial proof arising from this mapping, which is frequently required.

**Could you restate the problem? Could you restate it still differently? Go back to definitions.**

"Restating the problem" is a clear invitation to revise the problem representation in order to make it simpler to prove. Automated reasoning systems have made no use of such representation variations; they stick with the initial formalisation. The "definitions", of course, are where the fundamental representational decisions were made, e.g., how is this object to be idealized? So, going back to definitions is exactly what you have to do to reformalise the problem.

**If you cannot solve the proposed problem, try to solve first some related problem. Could you imagine a more accessible related problem? A more general problem? A more special problem? An analogous problem? Could you solve part of the problem? Keep only part of the condition, drop the other part; how far is the unknown then determined, how can it vary? Could you derive something useful from the data? Could you think of other data appropriate to determine the unknown? Could you change the unknown or the data, or both if necessary, so that the new unknown and the new data are nearer to each other?**

There's a lot going on in the above advice. Taken as a whole, it seems to suggest forming a related problem with a similar, but different, conjecture. In particular, we are invited to weaken and strengthen the original conjecture in various ways. The advantages of weakening are obvious; this is the classic abstraction approach, whereby a simpler conjecture is formulated and proved, then its proof is used to guide the proof of the original conjecture by analogy or the theorem is used as a key lemma. It's not so obvious why *strengthening* might help. However, generalisation of a theorem will often throw away unnecessary clutter, simplifying the task[5]. Especially in proofs by mathematical induction, generalisation will strengthen the induction hypothesis, providing a more powerful assumption to prove the similarly strengthened induction conclusion.

This advice falls somewhere between representational experimentation and proof search.

**Did you use all the data? Did you use the whole condition? Have you taken into account all essential notions involved in the problem?**

This is pure proof search — suggesting the prioritisation of assumptions and hypotheses.

## Carrying Out the Plan

The heading of this subsection sounds as if the advice in it is about the application of the proof plan to the problem, e.g. the running of the tactics, but closer inspection belies that interpretation.

**Carrying out your plan of the solution, *check each step*. Can you see clearly that each step is correct? Can you prove that it is correct?**

The steps of logic-based problem solvers are typically correct by construction, so this advice has little purchase from a proof-search perspective.

However, if the solution steps have not have been fully formalised, they may not be correct by construction. Moreover, we will have some additional reasoning to check: that the problem representation we have chosen is faithful to the original problem statement.

## Looking Back

Looking back is not normally part of automated reasoning. Once a proof is found the job is finished. There has been a limited amount of work on learning from proofs, e.g. learning new proof plans, tactics or derived rules from example proofs. This sounds related to improving proof search. However, again first appearances are deceptive.

**Can you *check the result?* Can you check the argument?**

This seems to be the same advice as that given in the last subsection. Similar remarks apply.

[5]Lakatos also discusses the use of generalisation to patch a faulty conjecture by including some counter-examples to the original conjecture. The TM system implements some Lakatos-inspired methods for correcting faulty conjectures by specialising them to exclude counterexamples [Colton & Pease, 2004], but it does not use specialisation as a route to prove an original correct conjecture.

**Can you derive the result differently? Can you see it at a glance?**

Problem solvers usually work in a search space of possible solutions. It is technically simple to arrange for this space to be searched for more than one solution, although it is unclear what this gains unless you need a solution with some special property, e.g., optimality, efficient execution, reliability.

However, if a different *representation* of the problem yields the same result, then this may give us faith in the robustness of the solution to representational variations.

**Can you use the result, or the method, for some other problem?**

This advice relates to that in §. In particular, it is the flip side of proof by analogy: making solutions available for recycling. Again, I remark that this concerns remembering good representational techniques at least as much as it does proof-search ones.

# Lakatos's Proofs and Refutations

Imre Lakatos (1922-1974) was a philosopher of mathematics and science. At the London School of Economics he worked with Karl Popper, and was influenced to apply Popper's ideas on the philosophy of science to mathematics. His controversial theory was that, as in science, the methodology of mathematics had evolved over time, in particular, that its ability to deal with fallible conjectures and their counter-examples had gradually become more sophisticated. In his book "Proofs and Refutations", he provides an accessible account of this theory via a rational reconstruction of the history of Euler's theorem about polyhedra. Lakatos was also influenced by Pólya. For instance, it was Pólya who suggested using Euler's Theorem as the vehicle for Lakato's work. Some of the methodological techniques described by Lakatos are closely related to Pólya's advice.

Polyhedra are the 3D version of polygons: objects whose faces are polygons. Examples include the five regular Platonic 'solids'[6]: tetrahedron, cube, octahedron, dodecahedron and icosahedron, as well as many other semi-regular and regular objects. I don't want to be more specific at this stage for reasons that will become apparent.

Lakatos describes a fictional classroom in which a teacher leads a (very bright!) class through this history. The teacher starts by presenting Cauchy's proof of the theorem. The students then confront it with various counterexamples and suggest ways to cope with them. The methodological techniques become more sophisticated as we progress through the book. In particular, there are refinements of definitions, of proofs and of proof analysis techniques. In her PhD project, Alison Pease implemented Lakatos's methods of surrender, exception-barring, piecemeal withdrawal, monster-barring and lemma-incorporation. These were implemented in her TM [Colton & Pease, 2004] and HRL [Pease *et al*, 2009] systems. We will meet some of these methods in the discussion below.

I recommend anyone who has not read [Lakatos, 1976] to do so. It is short, accessible and educational, but above all fascinating and great fun!

## Euler's Theorem and Cauchy's 'Proof' of It

Euler's theorem is that $V - E + F = 2$, where $V$ is the number of vertices of a polyhedron, $E$ is the number of its edges and $F$ is the number of its faces. The teacher presents the following proof couched as a "thought experiment", quoted from [Lakatos, 1976][p7-8].

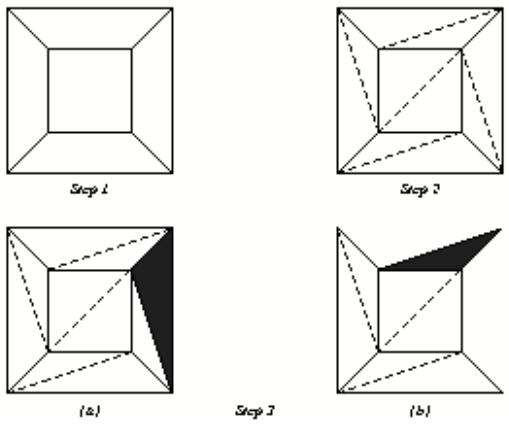
**"Step 1** Let us imagine the polyhedron to be hollow, with a surface made of thin rubber. If we cut out one of the faces, we can stretch the remaining surface flat on the blackboard, without tearing it. The faces and edges will be deformed, the edges may become curved, but $V$ and $E$ will not alter, so that if and only if $V - E + F = 2$ for the original polyhedron,

[6]The reason for the scare quotes around "solid" will soon become apparent.

$V - E + F = 1$ for this flat network – remember we have removed one face (Figure 1 top left shows the flat network for the case of a cube).

**"Step 2** Now we triangulate our map – it does indeed look like a geographical map. We draw (possibly curvilinear) diagonals in those (possibly curvilinear) polygons which are not already (possibly curvilinear) triangles. By drawing each diagonal we increase both $E$ and $F$ by one, so that the total $V - E + F$ will not be altered (Figure 1 top right).

**"Step 3** From the triangulated network we now remove the triangles one by one. To remove a triangle we either remove an edge – upon which one face and one edge disappear (Figure 1 bottom left), or we remove two edges and a vertex – upon which one face, two edges and one vertex disappear (Figure 1 bottom right). Thus if $V - E + F = 1$ before a triangle is removed, it remains so after the triangle is removed. At the end of this procedure we get a single triangle. For this $V - E + F = 1$ holds true. Thus we have proved our conjecture."



*Each of the four diagrams shows successive steps in the application of the schematic proof to a cube. First it is stretched onto the plane. Then it is triangulated. Then triangles are removed. The two cases of triangle removal are shown.*

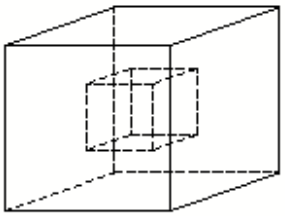
Figure 1: A Worked Example of the Proof

Notice that this proof is a procedure, similar to a computer program. Given a particular polyhedron, this procedure will produce a proof of Euler's Theorem, *customised for this polyhedron.* Notice, in particular, that the number and nature of the operations in Steps 2 and 3 will vary depending on the polyhedron they are applied to. This kind of proof is called *schematic.* Although schematic proofs are unlike standard logical proofs, they can be automated (see, for instance, the PhD projects of Siani Baker [Baker *et al*, 1992] and Mateja Jamnik [Jamnik *et al*, 1997]) using a logical rule called the *constructive omega rule.* These implementations start by working out the steps required in one or more concrete examples, treat these steps as the trace of the required procedure, generalise the trace to induce this general procedure, and then optionally verify that this procedure will produce a correct proof whatever example it is applied to. In his MSc project, Andy Fugard showed that human informal reasoning often takes the form of schematic proof rather than standard logical proof [Fugard, 2005]. Schematic proof is not the main topic of this paper, but an overview and discussion of its importance can be found in [Bundy *et al*, 2005].

## Definitions can *Follow* Proofs

Before long, [Lakatos, 1976][p13], one of the students has a counterexample. This consists of a solid cube containing a cube shaped hole, for which $V - E + F = 4$ (Figure 2). The first reaction is to

just give up the theorem[7]. However, a second reaction soon emerges to disbar the counterexample by ruling it not a polyhedron. Lakatos calls this *the method of monster barring.*



*A cube containing a cube shaped hole. Note that* $V - E + F = 16 - 24 + 12 = 4$. *Is this a polyhedron?*

Figure 2: The Hollow Cube

Two rival definitions of polyhedron are proffered:

1. "A polyhedron is a solid whose surface consists of polygonal faces."
2. "A polyhedron is a surface consisting of a system of polygons."

Under definition 1 the hollow cube is a polyhedron and under definition 2 it isn't. So by adopting definition 2 the hollow cube is disbarred as a counterexample and the theorem is saved.

Alternative definitions can be proposed because up to now the concept of polyhedron remained undefined. You may ask how it was possible to have a proof about an undefined concept. In logical theories, formal definitions precede proofs. Here, it is the other way around. The use of *schematic* proofs makes this possible. A procedure can be defined without specifying *exactly* what kind of inputs (in this case polyhedra) it will be given. Only later do we discover that it fails to apply to certain inputs (in this case we can't apply Step 1 to the hollow cube). This kind of experience is commonplace in practical computer programming, and is a common source of program bugs[8].

Note also that definitions 1 and 2 are not totally formal. They call upon other concepts: "solid", "surface', "polygon", "consists of", "bounded", "system", *etc*, which must also be defined. If any of these is informal, i.e., has grey areas itself, then we have merely postponed the problem. This greyness immediately becomes apparent when definition 2 is challenged in [Lakatos, 1976][p15] by two counterexamples consisting of two tetrahedron joined at a vertex or edge (Figure 3). This is resisted by tightening up the definition of "system" to exclude such combinations.

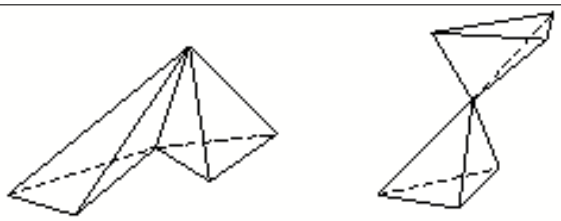
This kind of definition refinement was emulated in Pease's HRL system by having a variety of different kinds of definition and moving between them. Initially, for instance, a concept may only be 'defined' in an extensional way as a set of known examples of that concept, e.g., a set of known prime numbers. Later, this might be replaced by an intensional definition as a logical formula, e.g., prime numbers as numbers with only two divisors. Note that an intensional definition might refer to concepts that are only defined extensionally. During the emulation of the various Lakatos methods, these definitions might change in both directions and to different extensional and intensional definitions.

## Alternative Idealizations

Some counterexamples do not need to be disbarred, but can be turned into examples by redefining their properties. This method of monster adjustment is proposed in [Lakatos, 1976][p30]. Kepler's

[7]The method of surrender.

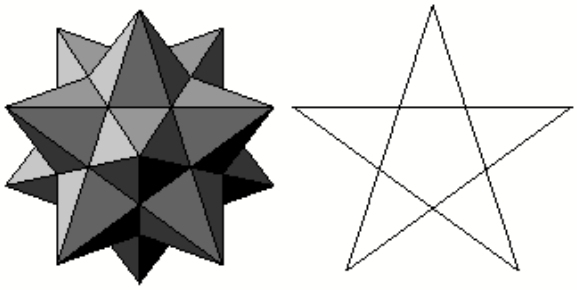
[8]Note that some programming languages preclude this kind of error by requiring the programmer to formally define the type of inputs to any program.



*Each composite polyhedron consists of two tetrahedron joined at an edge (on the left) and a vertex (on the right). For the left hand one:* $V - E + F = 6 - 11 + 8 = 3$ *and for the right hand one:* $V - E + F = 5 - 12 + 8 = 1$. *Are each of these really* one *polyhedron?*

Figure 3: Joined Tetrahedra

star-polyhedron occurs as a counterexample on p17 (Figure 4). It is considered to have 12 faces each of which is a five point star. These faces intersect. Under this definition it has 12 vertices and 30 edges, so that $V - E + F = -6$.



*On the left is a small stellated dodecahedron, also called Kepler's star-polyhedron and the "urchin". Each of its faces is a pentagram, seen on the right, which has five sides. It has 12 such faces, forming 12 vertices and 30 edges,* i.e. $V - E + F = 12 - 30 + 12 = -6$. *It can also be seen as containing 60 triangular faces. This time* $V - E + F = 32 - 90 + 60 = 2$*, as desired. Which is the right interpretation, or are both correct?*

Figure 4: Kepler's Star-Polyhedron

However, on p31 the faces of this star-polyhedron are redefined to be the triangles at the points of the star-polygons. This gives 60 ($5 \times 12$) faces, 90 edges and 32 vertices, so $V - E + F = 2$, as required.

If we regard the star-polyhedron as being defined as a set of cartesian coordinates the definitions of what constitute faces, edges or vertices are somewhat similar to the vision algorithms which find objects in pixel arrays. We should not, therefore, be surprised if minor variants of such definitions return different answers. For instance, does the search for a face stop when we reach an edge or can it carry on through the object to match up with another co-planar face on the other side? This is what determines whether we "see" 12 star shaped faces in the Kepler star-polyhedron or 60 triangular ones.

Maybe, as one of the "students" in [Lakatos, 1976][p33] claims, it can be both, i.e., the same

set of points can be viewed as two distinct polyhedra. This forces us to distinguish the object viewed as a set of points (of which there is one) from the object viewed as a polyhedron (of which there are two). This example shows us there is often several ways of idealizing an informal object as a formal one. We explored such choices in our MECHO project to solve mechanics problems stated in English [Bundy *et al*, 1979]. For instance, in a lever problem, a man carrying a ladder might play the role of the fulcrum of the lever, whereas the same man standing on a scaffold board might play the role of a weight.

## New Concepts from Repairing Faulty Conjectures

Another method of dealing with counterexamples is to modify the theorem to incorporate some conditions into it. This is discussed in [Lakatos, 1976][p33]. Euler's theorem is reformulated as one not about all polyhedra, but only *simple polyhedra*. A simple polyhedron is one which can be stretched onto the plane, which is step 1 of the schematic proof in §, *i.e.* those that are topologically equivalent to a sphere. None of the counterexamples is a simple polyhedron, so this excludes them. Lakatos calls this the *method of lemma incorporation* because he regards steps 1-3 as lemmas and he is ensuring the truth of one of these lemmas by adding a condition to the theorem.

Again this revision requires evolving representations — this time the creation of a new definition for the new concept of simple polyhedra. What is novel is the method of discovering the concept and its definition. The failed proof is analysed with the aid of a counterexample to see at what point the proof fails. A new precondition is then introduced into the faulty conjecture to ensure that this failure does not occur. For instance, step 1 of the proof cannot be carried out for any of the counterexamples, so we restrict the proof to polyhedra for which, *by definition*, step 1 *can* be carried out. The technique of repairing a faulty conjecture by constructing a new precondition for it by analysing a failed proof attempt is called *precondition analysis*. It has been reinvented and implemented several times in the history of automated reasoning. The earliest occurrence I am aware of is in J Moore's PhD thesis [Moore, 1974][p147]. In her implementation of Lakatos's methods, Alison Pease used Colton's HR concept formation program to create new concepts to refine faulty conjectures [Colton & Pease, 2004, Pease *et al*, 2009].

Soon counterexamples are found which fail at other steps of the proof and which require further conditions to be built into the theorem statement, e.g., that faces be *simply connected* in order that step 2 be applicable. Sometimes the assumptions in the proof that are violated are implicit, so that the counterexample first reveals the hidden assumption before supplying the extra condition in the theorem. This situation is maybe analogous to writing a program for integers but implicitly assuming they were positive integers. We only notice when either (a) we apply the program to a negative integer and it goes wrong or (b) we try to verify that the program meets a formal specification and find the proof attempt breaks down.

One might worry that there is no bottom to this process; that we might go on discovering counterexamples and, hence, hidden assumptions forever; that we will be forced to add new conditions to the theorem, refining our representations indefinitely. One of the students in [Lakatos, 1976][p40] expresses this worry explicitly as "an infinite regress in proofs". In 20th century mathematics we seem to have bottomed out in various axiom sets and logical rules of inference, but there is always the potential to unpack these further by deriving existing axioms and rules in terms of yet more primitive ones. The potential for bottoming out is a characteristic of mathematical reasoning which distinguishes it from commonsense reasoning. It arises because in mathematics we are reasoning about ideal objects for which we can agree a formal definition. In real life the representation is only a model of reality – a model which we can constantly refine to show more detail.

Notice again how representations are still evolving, with the addition of new concepts, *after* the initial (faulty) proof has been formulated.

## Counterexamples from Proof Analysis

Not only can counterexamples be used to analyse and modify proofs, but the analysis of proofs can be used to suggest counterexamples. By locating the assumptions in a proof we can design counterexamples which violate these assumptions.

Lakatos calls this interplay between proofs and counterexamples the *method of proofs and refutations*. At its heart is the technique of proof analysis. Proofs are examined in detail to identify concepts which are vague and assumptions which are made. Counterexamples are taken through each proof step in turn to see which ones they violate and so assist proof analysis. The result is a strengthening of definitions and proofs.

## New Concepts from Theorem Generalisation

Adding new preconditions to a theorem weakens its content, *e.g.* Euler's theorem covers a narrower class of polyhedra. To counteract this we need to find ways to generalise the theorem.

One way to do this is by proof analysis using examples which violate one of the "lemmas", i.e., proof steps, but for which the theorem is still true. Lakatos calls these *local* but not *global* counterexamples. Is there some way to weaken the conditions, e.g., by replacing them with disjunctions, and hence allow the theorem to apply to more polyhedra. [Lakatos, 1976][p57] mentions this possibility, but fails to find any examples of it.

Another route to generalisation is to find a theorem which accounts for the global counterexamples as well. In the case of Euler's theorem we want a formula which gives a value for $V-E+F$ for all polyhedra. [Lakatos, 1976][p80-1] gives a series of more general such formulae, culminating in:

$$V - E + F \quad = \quad \sum_{j=1}^{K} \{2 - 2(n_j - 1) + \sum_{k=1}^{F} e_{kj}\}$$

where $K$ is the number of disconnected surfaces, the $jth$ surface is an $n_j$-spheroid polyhedron and $e_{kj}$ is the number edges in each face which can be deleted without reducing the number of faces. This formula requires the invention and definition of three new concepts: "disconnected surface", "$n$-spheroid polyhedron" and "number edges in each face which can be deleted without reducing the number of faces". Inventing these concepts requires proof analysis using the global counterexamples to identify the properties that cause them to violate the proof and to discover what alternative value the proof gives for $V - E + F$ when these properties are present.

This example provides yet another example of the way in which representation evolution, including theorem formation and the introduction of new concepts, is interleaved with problem solving using the current state of the representation.

## Formalization of Informal Concepts

In [Lakatos, 1976][chapter 2] is a modern formal proof of Euler's theorem. This works by representing polyhedra as sets of vertices, edges and faces together with incidence matrices to say which vertices are in each edge and which edges are in each face. A restricted class of polyhedra is defined using concepts from this representation. The theorem for this restricted class of polyhedra is then turned into a formulae of vector algebra and a calculation in this algebra gives the value 2 for $V - E + F$, as required.

This theorem is claimed to be irrefutable, i.e., no counterexamples to it are possible. This is because it is proved by a formal derivation from axioms. Proof analysis cannot find any assumptions or vague definitions in the proof, so there is no room for counterexamples to be constructed.

Modulo the possibility of careless slips in applying rules of inference, we can accept this claim. However, the possibility of error has now shifted from the proof to the modelling process. We are asked to accept on trust that the formal definition of polyhedra coincides with our intuitive one; that the restricted class of polyhedra coincides with the intuitively simple polyhedra.; and that the

axioms and rules of our formal theory are true for polyhedra. This is not a trivial matter. Some of the formal definitions and axioms are quite obtuse and some thought experiments are required to see that they coincide with our intuitions. These thought experiments are not dissimilar to the thought experiment which underlies the schematic proof in §.

The process of associating a formal representation with an informally stated problem was explored in our Mecho, [Bundy *et al*, 1979], and Eco, [Robertson *et al*, 1991], projects. It is highly non-deterministic. We must choose a formal theory and then find formal counterparts in it for the objects and relations in the informal problem. At each stage we have a lot of choice. Some aspects of the informal problem can be neglected as negligible. Objects and relations can be represented with more or less complexity. In areas like mechanics the ground rules for doing this are well established by centuries of practice. In ecology there are fewer constraints. In all cases the decisions depend on circumstances, e.g., how accurate an answer is required; how much calculation are we prepared to do; which aspects of the problem are most important? It is always possible to model the real world in more detail. Do we take account of friction? What about the pressure of the solar wind? Are these particles to be regarded as one object, divided into clumps or considered individually? This is where the bottomless regress reenters.

# Conclusion

What lessons can we draw from this analysis?

1. If, as I claim above, Pólya's problem solving advice is mainly geared to the choice, construction and evolution of representations best suited to efficient problem solving, and if research on autonomous reasoning agents has been mainly geared to guiding proof search, then this would help explain why the take-up of his advice by the automated reasoning community has been essentially non-existent.

2. However, the autonomous reasoning agents of the future will need to treat their representations as fluid, as much as they currently derive new conclusions from old ones. In this world, Pólya's advice might prove much more useful.

3. Lakatos's illustrations of mathematical methodology teach us that we need not wait for representation to be complete before reasoning can start. Reasoning can start with an incomplete representation and some of its results may be to develop and refine the representation, as well as reasoning with the current representations.

4. Initially, concepts may be represented in an informal manner which allows some grey area, i.e., it may not be clear whether a new object fits the concept or not. An autonomous agent may choose whether it does and, hence, refine its concept accordingly.

5. A concept's definition may depend on undefined concepts. These, in their turn, may later be defined in terms of other undefined concepts. In commonsense reasoning this regress may be bottomless, but in mathematics it usually bottoms-out in some *atomic* concepts. This decision may be revisited in some later representation in which definitions are given for previously atomic concepts.

6. New concepts can emerge, for instance, from attempts to patch a failed proof and to generalise a conjecture. For instance, it may be necessary to introduce a new precondition to a conjecture to ensure its correctness. The formal definitions of these new concepts may come later in the process.

7. Part of representation formation is idealization: the assignment of formal roles and definitions to informal objects. There is often a choice about how this is to be done — a choice that may need to be remade as part of the problem-solving process.

By showing that Pólya's advice and Lakatos's analysis are as much, if not more, about representation than about reasoning, we explain why, despite their popularity, they have played such a small role in the automation of reasoning. We also predict that they will play a much bigger role in the automation of *fluid* representations for autonomous agents.

# References


[Baker *et al*, 1992] Baker, S., Ireland, A. and Smaill, A. (1992). On the use of the constructive omega rule within automated deduction. In Voronkov, A., (ed.), *International Conference on Logic Programming and Automated Reasoning — LPAR 92, St. Petersburg*, Lecture Notes in Artificial Intelligence No. 624, pages 214–225. Springer-Verlag.

[Bundy & McNeill, 2006] Bundy, A. and McNeill, F. (2006). Representation as a fluent: An AI challenge for the next half century. *IEEE Intelligent Systems*, 21(3):85–87.

[Bundy, 1991] Bundy, A. (1991). A science of reasoning. In Lassez, J.-L. and Plotkin, G., (eds.), *Computational Logic: Essays in Honor of Alan Robinson*, pages 178–198. MIT Press.

[Bundy, 2006] Bundy, A. (2006). Constructing, selecting and repairing representations of knowledge. In *Proceedings of ai@50: The Dartmouth Artificial Intelligence Conference: The next fifty years*. Invited Talk.

[Bundy *et al*, 1979] Bundy, A., Byrd, L., Luger, G., Mellish, C., Milne, R. and Palmer, M. (1979). Solving mechanics problems using meta-level inference. In Buchanan, B. G., (ed.), *Proceedings of IJCAI-79*, pages 1017–1027. International Joint Conference on Artificial Intelligence.

[Bundy *et al*, 2005] Bundy, A., Jamnik, M. and Fugard, A. (2005). What is a proof? *Phil. Trans. R. Soc A*, 363(1835):2377–2392.

[Colton & Pease, 2004] Colton, Simon and Pease, Alison. (2004). The TM system for repairing non-theorems. In Ahrendt, W., Baumgartner, P., de Nivelle, H., Ranise, S. and Tinelli, C., (eds.), *Selected papers from the IJCAR'04 disproving workshop*, volume 125 (3) of *Electronic Notes in Theoretical Computer Science*, pages 87–101.

[Fugard, 2005] Fugard, A.J.B. (2005). An exploration of the psychology of mathematical intuition. Unpublished M.Sc. thesis, School of Informatics, Edinburgh University.

[Jamnik *et al*, 1997] Jamnik, Mateja, Bundy, Alan and Green, Ian. (1997). Automation of diagrammatic reasoning. In Pollack, M.E., (ed.), *Proceedings of the 15th IJCAI*, volume 1, pages 528–533. International Joint Conference on Artificial Intelligence, Morgan Kaufmann Publishers. Also published in the "Proceedings of the 1997 AAAI Fall Symposium". Also available from Edinburgh as DAI Research Paper No. 873.

[Lakatos, 1976] Lakatos, I. (1976). *Proofs and Refutations: The Logic of Mathematical Discovery*. Cambridge University Press.

[Moore, 1974] Moore, J S. (1974). *Computational Logic: Structure sharing and proof of program properties, part II*. Unpublished Ph.D. thesis, University of Edinburgh, Available from Edinburgh as DCL memo no. 68 and from Xerox PARC, Palo Alto as CSL 75-2.

[Owen, 1990] Owen, S. (1990). *Analogy for Automated Reasoning*. Academic Press Ltd.

[Pease *et al*, 2009] Pease, A., Smaill, A., Colton, S. and Lee, J. (2009). Bridging the gap between argumentation theory and the philosophy of mathematics. *Foundations of Science*, 14(1-2):111–135.

[Pólya, 1945] Pólya, G. (1945). *How to Solve It*. Princeton University Press.

[Robertson *et al*, 1991] Robertson, D., Bundy, A., Muetzelfeldt, R., Haggith, M. and Uschold, M. (1991). *Eco-Logic: Logic-Based Approaches to Ecological Modelling*. MIT Press.

[Sutcliffe & Puzis, 2007] Sutcliffe, G. and Puzis, Y. (2007). SRASS: a semantic relevance axiom selection system. In *Proc. of CADE*.

[Winterstein *et al*, 2002] Winterstein, D., Bundy, A., Gurr, C. and Jamnik, M. (2002). Using animation in diagrammatic theorem proving. 46-60. In Hegarty, Mary, Meyer, Bernd and Narayanan, N. Hari, (eds.), *Diagrams 2002*, volume 2317 of *Lecture Notes in Computer Science*, pages 46–60. Springer.